

✓ 67531-US
KK/mk.

日本国特許庁
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出願年月日 2002年12月13日
Date of Application:

出願番号 特願2002-362487
Application Number:

[ST. 10/C]: [JP 2002-362487]

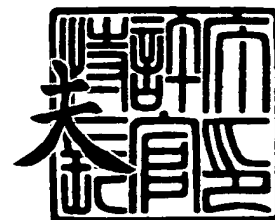
出願人 株式会社デンソー
Applicant(s):



2003年 8月18日

特許庁長官
Commissioner,
Japan Patent Office

今井 康



出証番号 出証特2003-3067230

【書類名】 特許願

【整理番号】 PNID3972

【提出日】 平成14年12月13日

【あて先】 特許庁長官殿

【国際特許分類】 G06F 15/00

【発明者】

 【住所又は居所】 愛知県刈谷市昭和町 1 丁目 1 番地 株式会社デンソー内

 【氏名】 宮本 誠司

【発明者】

 【住所又は居所】 愛知県刈谷市昭和町 1 丁目 1 番地 株式会社デンソー内

 【氏名】 井原 博之

【特許出願人】

 【識別番号】 000004260

 【氏名又は名称】 株式会社デンソー

【代理人】

 【識別番号】 100082500

 【弁理士】

 【氏名又は名称】 足立 勉

 【電話番号】 052-231-7835

【手数料の表示】

 【予納台帳番号】 007102

 【納付金額】 21,000円

【提出物件の目録】

 【物件名】 明細書 1

 【物件名】 図面 1

 【物件名】 要約書 1

 【包括委任状番号】 9004766

【プルーフの要否】 要

【書類名】 明細書

【発明の名称】 車両用制御プログラム、車両用制御装置、車両用制御プログラムの生成方法

【特許請求の範囲】**【請求項 1】**

プラットフォームプログラムと、アプリケーションプログラムとを備える車両用制御プログラムにおいて、さらに結合処理プログラムを備え、

前記プラットフォームプログラムは、ハードウェアデバイスからデータを入力し、異なる車両用制御装置の要求仕様に従って作成されたアプリケーションプログラムによる処理によって共通に利用可能なように標準化したインターフェースである P F インターフェースに従って、前記入力したデータに基づくデータを前記結合処理プログラムによる処理に提供する処理をコンピュータに実行させるためのプログラムであり、

前記結合処理プログラムは、前記 P F インターフェースに従って前記プラットフォームプログラムによる処理から提供されたデータを、開発対象の車両用制御装置の要求仕様を満たすインターフェースである A P インターフェースに適合するように変換することで前記アプリケーションプログラムによる処理に仲介する処理をコンピュータに実行させるためのプログラムであり、

前記アプリケーションプログラムは、前記 A P インターフェースに従って前記結合処理プログラムによる処理から提供されたデータを用いた処理をコンピュータに実行させるためのプログラムであること

を特徴とする車両用制御プログラム。

【請求項 2】

請求項 1 に記載の車両用制御プログラムにおいて、

前記 P F インターフェースとして標準化する対象は、当該 P F インターフェースを介して提供するデータの構造及び当該データの更新タイミングであること
を特徴とする車両用制御プログラム。

【請求項 3】

請求項 1 または 2 に記載の車両用制御プログラムにおいて、

前記プラットフォームプログラムは、前記 P F インターフェースを介して提供するデータを、前記アプリケーションプログラムによる処理で要求される精度よりも高い精度のデータ形式で提供する処理をコンピュータに実行させるためのプログラムを備え、

前記結合処理プログラムは、前記アプリケーションプログラムの要求仕様を満たすデータ形式となるように、前記 P F インターフェースを介して提供されたデータのデータ形式を調整して前記 A P インターフェースに従って提供する処理をコンピュータに実行させるためのプログラムを備えること

を特徴とする車両用制御プログラム。

【請求項 4】

請求項 1～3 のいずれかに記載の車両用制御プログラムにおいて、

前記プラットフォームプログラムは、前記 P F インターフェースに従って提供するデータを、前記アプリケーションプログラムによる処理で要求されるよりも高いサンプリングタイミングで提供する処理をコンピュータに実行させるためのプログラムを備え、

前記結合処理プログラムは、前記要求仕様を満たすデータのサンプリングタイミングへ、前記 P F インターフェースを介して提供されたデータの提供タイミングを調整し、前記 A P インターフェースによって提供する処理をコンピュータに実行させるためのプログラムを備えること

を特徴とする車両用制御プログラム。

【請求項 5】

請求項 1～4 のいずれかに記載の車両用制御プログラムにおいて、

前記 A P インターフェースは、前記変換したデータを、前記アプリケーションプログラムによる処理によって参照可能に構成したものであること

を特徴とする車両用制御プログラム。

【請求項 6】

請求項 1～5 のいずれかに記載の車両用制御プログラムにおいて、

前記 A P インターフェースは、前記アプリケーションプログラムによる処理によってデータの要求があった場合に、前記変換したデータを前記アプリケーショ

ンプログラムによる処理に渡すように構成したものであること
を特徴とする車両用制御プログラム。

【請求項 7】

請求項 1 ～ 6 のいずれかに記載の車両用制御プログラムにおいて、
前記結合処理プログラムは、前記 P F インターフェースを介して取得したデータまたは前記変換したデータを記憶手段に記憶しておき、当該記憶手段に記憶されたデータを変換したデータあるいは当該記憶手段に記憶されたデータを前記 A P インターフェースを介して提供するプログラムを備えること
を特徴とする車両用制御プログラム。

【請求項 8】

請求項 7 に記載の車両用制御プログラムにおいて、
前記プラットフォームプログラムは、前記ハードウェアデバイスからの割り込みによって、前記ハードウェアデバイスからのデータの入力を前記コンピュータに行わせるための割り込み処理プログラムを備え、前記割り込み処理プログラムは、当該入力結果に基づくデータを前記 P F インターフェースを介して提供する処理をコンピュータに実行させるためのプログラムを備え、
前記結合処理プログラムは、さらに、他の処理のディスパッチを禁止している間に、前記 P F インターフェースを介してデータを受け取り、前記記憶手段に記憶する処理をコンピュータに実行させるためのプログラムである取得プログラムを備えること
を特徴とする車両用制御プログラム。

【請求項 9】

プラットフォームプログラムと、アプリケーションプログラムとを備える車両用制御プログラムにおいて、さらに結合処理プログラムを備え、
前記プラットフォームプログラムは、異なる車両用制御装置の要求仕様に従って作成されたアプリケーションプログラムによる処理によって共通に利用可能なように標準化したインターフェースである P F インターフェースに従って出力のためのデータを前記結合処理プログラムによる処理から取得し、取得したデータに基づくデータを出力する処理をコンピュータに実行させるためのプログラムで

あり、

前記結合処理プログラムは、開発対象の車両用制御装置の要求仕様を満たすインターフェースである A P インターフェースに従って前記アプリケーションプログラムによる処理から提供されたデータを、前記 P F インターフェースに適合するように変換して、前記プラットフォームプログラムによる処理に仲介する処理をコンピュータに実行させるためのプログラムであり、

前記アプリケーションプログラムは、出力対象のデータを生成し、前記 A P インターフェースに従って前記出力対象のデータを前記結合処理プログラムによる処理に提供する処理をコンピュータに実行させるためのプログラムであること
を特徴とする車両用制御プログラム。

【請求項 10】

請求項 9 に記載の車両用制御プログラムにおいて、

前記 A P インターフェースは、前記アプリケーションプログラムによる処理によって記憶されたデータを結合処理プログラムによる処理によって自ら取得する構成とすること

を特徴とする車両用制御プログラム。

【請求項 11】

請求項 1 ～ 8 のいずれかに記載の車両用制御プログラムと請求項 9 または 10 に記載の車両用制御プログラムとを備える車両用制御プログラム。

【請求項 12】

請求項 1 ～ 11 のいずれかに記載の車両用制御プログラムと当該車両用制御プログラムを実行するコンピュータとを備えた車両用制御装置。

【請求項 13】

ハードウェアデバイスからデータを入力し、異なる車両用制御装置の要求仕様に従って作成されたアプリケーションプログラムによる処理によって共通に利用可能なように標準化したインターフェースである P F インターフェースに従って、前記入力したデータに基づくデータを前記結合処理プログラムによる処理に提供する処理をコンピュータに実行させるためのプログラムであるプラットフォームプログラムを生成し、

前記 P F インターフェースに従って前記プラットフォームプログラムによる処理から提供されたデータを、開発対象の車両用制御装置の要求仕様を満たすインターフェースである A P インターフェースに適合するように変換することで前記アプリケーションプログラムによる処理に仲介する処理をコンピュータに実行させるためのプログラムである結合処理プログラムを生成し、

前記 A P インターフェースに従って前記結合処理プログラムによる処理から提供されたデータを用いた処理をコンピュータに実行させるためのプログラムであるアプリケーションプログラムを生成し、

前記プラットフォームプログラムと、前記結合処理プログラムと、前記アプリケーションプログラムとをリンクして車両用制御プログラムを生成すること

を特徴とする車両用制御プログラムの生成方法。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

車両用制御プログラム、例えばエンジン制御用組込ソフトウェアとして好適な構成をもつプログラム等に関する。

【0002】

【従来の技術】

従来より、例えばマイコン等のコンピュータを用いて、自動車等の車両の各部を制御する車両用制御装置が知られている。こうした車両用制御装置では、ROM等の記憶装置に記憶された車両用制御プログラムをコンピュータが実行することにより、種々の制御を実現している。

【0003】

車両用制御プログラムは、一般的に、ハードウェアに依存するプログラムであるプラットフォームプログラム（P F）と、車両用の制御に依存するプログラムであるアプリケーションプログラム（A P）とに分けて構成する。

プラットフォームプログラムは、ハードウェアデバイスからデータを入力して、その入力したデータに基づくデータを、例えば A P I（Application Program Interface）のようなインターフェース（以下このインターフェースを P F イン

ターフェーズと称する) を介して、アプリケーションプログラムによる処理に対して提供したり、アプリケーションプログラムによる処理からこの P F インターフェースを介して得たデータをハードウェアデバイスに対して出力したりする処理をコンピュータに実行させるためのプログラムである。

【 0 0 0 4 】

一方、アプリケーションプログラムは、この P F インターフェースを介して入力したデータを用いた処理や、P F インターフェースを介して出力するデータを生成する処理をコンピュータに実行させるためのプログラムである。例えば P F インターフェースを介して入力したデータに基づいて P F インターフェースを介して出力するデータを生成する処理をコンピュータに実行させるためのプログラムである。

【 0 0 0 5 】

このように、プラットフォームプログラムは、ハードウェアデバイスに対するアクセスを行うプログラムを含む一方、アプリケーションプログラムは、直接ハードウェアデバイスへのアクセスを行うプログラムを含まず、P F インターフェースを介して、プラットフォームプログラムによる処理にハードウェアデバイスとの入出力をさせている。こうすることで、アプリケーションプログラムの開発者は、ハードウェアの詳細構成や制御方法を意識することなくアプリケーションプログラムを作成することができる。

【 0 0 0 6 】

また、従来、例えば、特許文献 1 に示すように、オペレーティングシステムとアプリケーションソフトとを接続するインターフェースソフトを設け、アプリケーションソフトにおいて必要な信号の演算をインターフェースソフトで行うようにすることで、ハードウェアに変更があった場合にインターフェースソフトを書き換えるだけで「アプリケーションソフトを」永続的に使用可能にした自動車用制御ユニットも知られている。

【 0 0 0 7 】

【特許文献 1】

特開平 0 7 - 0 4 0 7 9 4 号公報

【0008】

【発明が解決しようとする課題】

しかしながら、車両用制御装置のメーカでは、顧客である車両メーカから異なる要求仕様を持つ複数の車両用制御装置の開発を依頼されることが多く、アプリケーションプログラムはそれぞれの要求仕様に応じて新規に作成する必要がある。したがって、上述した特許文献1に記載のように「アプリケーションプログラムソフトを」流用して車両用制御プログラムを作成するのではなく、プラットフォームプログラムを、できるだけ流用して、車両用制御プログラムを構成することにより、開発コストを削減する手法が採られる。

【0009】

プラットフォームプログラムを流用するためには、例えば、異なる要求仕様間の共通項を抽出し、その共通項で標準化したPFインターフェースを提供するようにプラットフォームプログラムを構成する。例えば各要求仕様の中で、最も要求の厳しい要求仕様に合わせたデータ型式やデータ提供タイミング等でデータをアプリケーションプログラムによる処理へ提供するようにPFインターフェースの仕様であるPFインターフェース仕様を標準化する。

【0010】

例えば車両用制御装置がエンジン制御装置であって、エンジンの冷却水の水温を検出するための水温センサからの電圧値に基づく処理を行う場合の要求仕様として、ある車種のエンジン制御装置Aでは、この電圧値の分解能として1/256Vが要求仕様であるのに対し、別の車種のエンジン制御装置Bでは分解能として1/128Vが要求仕様とされている場合を考える。このような場合には、プラットフォームプログラムによる処理では両方の車種のエンジン制御装置の要求仕様を共に満たすように分解能の細かいエンジン制御装置Aに合わせた分解能1/256Vで、アプリケーションプログラムによる処理に対してPFインターフェースを提供するPFインターフェース仕様を策定し、このPFインターフェース仕様に沿ったPFインターフェースを提供するようにプラットフォームプログラムを構成する。こうして、異なる要求仕様を持つエンジン制御装置で同一のプラットフォームプログラムを流用することができる。

【0011】

しかしながら、エンジン制御装置 B のアプリケーションプログラムを作成する場合、要求仕様の分解能は 1/128V であるのに対し、プラットフォームプログラムによる P F インターフェースが提供する分解能は 1/256V であるため、アプリケーションプログラムの開発者は、プラットフォームプログラムによる P F インターフェース仕様の分解能と、開発対象の車両用制御装置の要求仕様の分解能とのギャップを埋めるための分解能変換処理プログラムをアプリケーションプログラムに組み込む必要がある。

【0012】

こうした要求仕様としては、分解能の他にも、データの種類、データの型、データの取り込みタイミング等、様々なものがあり、異なる要求仕様を持つ車両用制御装置において標準化されたプラットフォームプログラムを利用するアプリケーションプログラムを作成する場合、アプリケーションプログラムの開発者は、常にプラットフォームプログラムの提供する P F インターフェース仕様を念頭において要求仕様を実現する方法を考える必要があり、アプリケーションプログラムの開発者の負担が大きいといった問題がある。

【0013】

そこで本発明は、アプリケーションプログラムの開発者の負担を軽減することのできる車両用制御プログラム等を提供することを目的とする。

【0014】**【課題を解決するための手段及び発明の効果】**

上述した問題点を解決するためになされた請求項 1 に記載の車両用制御プログラムは、プラットフォームプログラムと、アプリケーションプログラムと、結合処理プログラムとから構成されている。

【0015】

そして、プラットフォームプログラムは、ハードウェアデバイスからデータを入力し、異なる車両用制御装置の要求仕様に従って作成されたアプリケーションプログラムによる処理によって共通に利用可能なように標準化したインターフェースである P F インターフェースに従って、前記入力したデータに基づくデータ

を結合処理プログラムによる処理に提供する処理をコンピュータに実行させるためのプログラムである。

【0016】

また、結合処理プログラムは、P F インターフェースに従ってプラットフォームプログラムによる処理から提供されたデータを、開発対象の車両用制御装置の要求仕様を満たすインターフェースであるA P インターフェースに適合するように変換することでアプリケーションプログラムによる処理に仲介する処理をコンピュータに実行させるためのプログラムである。

【0017】

そして、アプリケーションプログラムは、A P インターフェースに従って結合処理プログラムによる処理から提供されたデータを用いた処理をコンピュータに実行させるためのプログラムである。

このように、結合処理プログラムが、P F インターフェースと、A P インターフェースとの橋渡しを行うため、アプリケーションプログラムの開発者は、A P インターフェース仕様、すなわち、開発対象の車両用制御装置の要求仕様を満たすインターフェース仕様に沿ってアプリケーションプログラムを作成することができる。すなわち、アプリケーションプログラムの開発者は、P F インターフェース仕様を念頭において要求仕様を実現する方法を考える必要はなくなるので、アプリケーションプログラムの開発者の負担を軽減することができる。

【0018】

このようなP F インターフェースとして標準化する対象としては、例えば、請求項2に示すように、P F インターフェースを介して提供するデータの構造及び当該データの更新タイミングなどが挙げられる。異なる要求仕様に従って作成されたアプリケーションプログラムは、異なるデータの構造を持つデータを必要とし、データの必要となるタイミング等も異なる。例えば、データの構造としては、どの入力対象（例えばどのセンサ）からの、どのような精度の、どのようなデータ型式（データ構造）のデータなのかという事項などを標準化して提供する。

【0019】

また、例えば、請求項3に示す車両用制御プログラムによれば、プラットフォ

ームプログラムによる処理で、P F インターフェースとして標準化して提供するデータを、アプリケーションプログラムによる処理で要求される精度よりも高い精度のデータ形式で提供し、結合処理プログラムによる処理で、アプリケーションプログラムの要求仕様を満たすデータ型式となるように、P F インターフェースを介して提供されたデータのデータ型式を調整してA P インターフェースによって提供する。このようにすれば、アプリケーションプログラムの作成時にP F インターフェースにおけるデータがどのような精度やデータ型式となっているのかを意識しなくても、要求仕様に従ったデータを利用することができる。

【0020】

また、例えばデータのサンプリングタイミングも請求項4のようにして、結合処理プログラムによる処理で調整するように、結合処理部を構成するとよい。このようにすれば、アプリケーションプログラムの開発者は、アプリケーションプログラムの作成時に、プラットフォームプログラムによる処理におけるサンプリングタイミングがどのようなになっているのかを意識することなく、要求仕様に従ったタイミングでデータを利用するように構成することができる。

【0021】

そして、A P インターフェースは、例えば、請求項5あるいは請求項6に示すようにしてインターフェースを行うようにすることができる。すなわち請求項5に示すようにして、変換したデータを、アプリケーションプログラムによる処理によって参照可能に構成するか、あるいは、請求項6に示すようにして、アプリケーションプログラムによる処理によってデータの要求があった場合に、変換したデータをアプリケーションプログラムによる処理に渡すように構成することができる。例えば、結合処理プログラムによる処理で予め定められたメモリアドレスのメモリ等の記憶手段にデータを記憶し、アプリケーションプログラムによる処理でその記憶手段を参照するようにしたり、あるいは、例えばいわゆるA P Iのように、アプリケーションプログラムによる処理から結合処理部による処理に移行して、結合処理部による処理で例えばスタック等へ要求されたデータまたはデータの格納先のアドレス（ポインタ等）を格納させて、アプリケーションプログラムによる処理へ戻り、スタック等からデータまたはデータの格納先のアドレ

スを取り出すように構成することができる。なお、P F インターフェースもこうした A P インターフェースと同様の方法によってインターフェースを行うことができる。

【0022】

そして、結合処理プログラムは、P F インターフェースを介して取得して変換したデータをそのまま A P インターフェースを介して提供するように構成してもよいが、例えば請求項 7 に示すように、P F インターフェースを介して取得したデータを記憶手段に記憶しておき、その記憶手段に記憶されたデータを変換したデータを A P インターフェースを介して提供するプログラムとしたり、P F インターフェースを介して取得したデータを加工したデータを記憶手段に記憶しておき、その記憶手段に記憶されたデータを A P インターフェースを介して提供するプログラムとしたりするとよい。

【0023】

例えば請求項 8 に示すように、プラットフォームプログラムには、ハードウェアデバイスからの入力をハードウェアデバイスからの割り込みによって行い、その入力結果に基づくデータを P F インターフェースを介して提供する処理のためのプログラムを備え、結合処理プログラムは、さらに、他の処理のディスパッチを禁止している間に、P F インターフェースを介してデータを受け取り、記憶手段に記憶する処理をさせるための取得プログラムを備えるとよい。このようにすれば、取得部による「P F インターフェースを介してデータを受け取り、記憶手段に記憶する処理」を行っている間は、他の処理のディスパッチが禁止されるので、データの一貫性を保つことが容易になる。また、アプリケーションプログラムによる処理では、例えば、この記憶手段に記憶されたデータを A P インターフェース介して取得して利用することができる。

【0024】

以上、ハードウェアデバイスからデータの入力処理を行う場合の車両用制御プログラムの構成について説明したが、ハードウェアデバイスへ出力を行う場合にも、同様に構成するとよい。すなわち、入力処理の場合と同様に、車両用制御プログラムをプラットフォームプログラムと、結合処理部と、アプリケーションプ

ログラムとに分けて構成する。例えば、請求項 9 に示すようにプラットフォームプログラムは、異なる車両用制御装置の要求仕様に従って作成されたアプリケーションプログラムによる処理によって共通に利用可能なように標準化したインターフェースである P F インターフェースに従って出力のためのデータを前記結合処理プログラムによる処理から取得し、取得したデータに基づくデータを出力する処理をコンピュータに実行させるためのプログラムとし、結合処理プログラムは、開発対象の車両用制御装置の要求仕様を満たすインターフェースである A P インターフェースに従ってアプリケーションプログラムによる処理から提供されたデータを、P F インターフェースに適合するように変換して、プラットフォームプログラムによる処理に仲介する処理をコンピュータに実行させるためのプログラムとし、アプリケーションプログラムは、出力対象のデータを生成し、A P インターフェースに従って前記出力対象のデータを結合処理プログラムによる処理に提供する処理をコンピュータに実行させるためのプログラムとする。このようにすれば、アプリケーションプログラムの開発者は、P F インターフェースの仕様を意識することなく、単に A P インターフェース仕様に従ってデータを出力するようにプログラムを作成するだけでよい。すなわち、要求仕様に従ったアプリケーションプログラムを作成するだけでよい。そのため、アプリケーションプログラムの開発者の負担を軽減することができる。

【0025】

A P インターフェースの構成としては種々の構成を採りうるが、例えば、請求項 10 に示すようにして、A P インターフェースは、前記アプリケーションプログラムによる処理によって記憶されたデータを結合処理プログラムによる処理によって自ら取得する構成とするとよい。そして、例えば、取得したデータに基づくデータをプラットフォームプログラムによる処理に渡してハードウェアデバイスへの出力処理をさせるように構成するとよい。このようにすれば、アプリケーションでは、単に演算結果（例えば駆動量や駆動タイミング）のデータを求めて記憶手段に記憶しておくだけで、結合処理プログラムによる処理によって、そのデータが自動的に取得され、プラットフォームプログラムの仕様を満足する適切なタイミングやデータ型式で、プラットフォームプログラムによる処理へ仲介さ

れる。したがって、アプリケーションの開発者は、アプリケーションの開発をスムーズに行うことができる。

【0026】

なお、結合処理プログラムは、入力あるいは出力のいずれかのみについて仲介を行うように構成してもよいが、入力と出力の双方を仲介するようにするとよい。例えば、請求項11に示すように、請求項1～8のいずれかに記載の車両用プログラムと請求項9または10に記載の車両用制御用プログラムとを備える車両用制御プログラムとするとよい。このようにすれば、アプリケーションプログラムの開発者は、入出力の双方について、プラットフォームプログラムによる処理によって提供されるPFインターフェースの仕様を意識することなく、単に要求仕様に従ってアプリケーションプログラムを作成することができる。

【0027】

そして、請求項12に示すように、請求項1～11のいずれかに記載の車両用制御プログラムと、この車両用制御プログラムを実行するコンピュータとを備えた車両用制御装置を構成することができる。このような車両用制御装置は、アプリケーションの開発を効率的に行うことができるため、従来より、低コストでの開発が容易にできるようになる。

【0028】

なお、請求項1に記載の車両用制御プログラムは、例えば請求項13に示す生成方法によって、生成（製造）することができる。（請求項13は、物（プログラム）を生産する方法の発明である。）なお、請求項13に示す生成方法におけるプラットフォームプログラムと結合処理部とアプリケーションの生成の順序はどのような順序でも構わない。

【0029】

例えば、新たな開発対象の車両用制御装置がある場合、プラットフォームプログラムについては、すでに開発済みの車両用制御装置のものをできるだけ再利用することとし、車両メーカーからの要求仕様を満足しない部分のみを修正して、PFインターフェース仕様を決定する。そして、車両メーカーからの要求仕様に従って、APインターフェースの仕様であるAPインターフェース仕様を決定する。

このように P F インターフェース仕様と A P インターフェース仕様を、新たな開発対象の車両用制御装置の開発の初期段階で決定すれば、プラットフォームプログラムと結合処理部とアプリケーションプログラムとを並行して開発することも可能となる。(なお、請求項 2 ~ 12 に記載の車両用制御プログラムについても、請求項 1 に対する請求項 13 の記載のようにして、生成方法の請求項とすることも可能である。)

【0030】

【発明の実施の形態】

以下、本発明が適用された実施例について図面を用いて説明する。なお、本発明の実施の形態は、下記の実施例に何ら限定されることなく、本発明の技術的範囲に属する限り種々の形態を採りうることは言うまでもない。

【0031】

本実施例の車両用制御装置は、エンジンの制御を行うためのエンジン ECU であり、図 1 に本実施例の説明に必要な部分についての構成を図示している。本実施例のエンジン ECU 10 と従来のエンジン ECU との差異は、ROM 12 に記憶されたプログラムの構成にあり、その他の構成は従来のエンジン ECU と同様である。

【0032】

エンジン ECU 10 は、図 1 に示すように、CPU 11、ROM 12、RAM 13、I/O 14、A/D 変換器 15、タイマ 16 等とこれらを接続するバスライン等から構成されている。そして、I/O 14 や A/D 変換器 15 には、エンジン回転センサ 21、O₂ センサヒータ 22、水温センサ 23、…、といった各種のエンジン制御用装置 20 が駆動回路等を介して接続されている。そして、ROM 12 に記憶されたプログラムを CPU 11 が実行することによって、こうしたエンジン制御用装置 20 からの信号の入力や、エンジン制御用装置 20 に対する信号の出力を行う。例えば、CPU 11 による処理によって、エンジン回転センサ 21 からのクランク軸の所定回転角毎 (例えば 180 度毎) に発生するパルスを I/O 14 から入力したり、I/O 14 に対して O₂ センサを所定の動作温度まで加熱するための O₂ センサヒータ 22 を PWM 制御する O₂ センサヒータ

制御信号を出力したりする。また、A/D変換器15を介して、水温センサ23から水温に対応する電圧値を入力し、入力した電圧値をデジタルデータとしてRAM13等へ取り込む。

【0033】

こうした制御をCPU11に実行させるためにROM12に記憶するプログラム1は、図2(a)に示すように、プラットフォームプログラム2(図中PFで表記している)と、特許請求の範囲における結合処理プログラムに相当する結合処理部3と、アプリケーションプログラム4(図中APで表記している)とをリンクして構成している。

【0034】

プラットフォームプログラム2は、I/O14やA/D変換器15等のハードウェアデバイスを制御する処理(ハードウェアに依存する処理)をCPU11が実行するためのデバイスドライバを含むプログラムであり、一方アプリケーション4は、I/O14やA/D変換器15といったハードウェアデバイスを直接制御する処理を行わず、車両メーカーからの要求仕様を実現するための処理(車両用の制御に依存する処理(判定処理、演算処理など))をCPU11が実行するためのプログラムである。そして、結合処理部3は、アプリケーションプログラム4によるCPU11の処理とプラットフォームプログラム2におけるCPU11の処理との仲介を行うための処理をCPU11が実行するためのプログラムである。

【0035】

プラットフォームプログラム2と結合処理部3とアプリケーションプログラム4は別々に開発が行われており、プラットフォームプログラム2は、すでに開発済みの従来のエンジンECUに用いられていたものを利用する。一方、結合処理部3とアプリケーションプログラム4は新規に開発する。

【0036】

プラットフォームプログラム2は、図2(b)に示すように、結合処理部3による処理からのI/O14やA/D変換器15の駆動要求を受け付けてI/O14やA/D変換器15を駆動し、また、I/O14やA/D変換器15から取得

した情報を結合処理部 3 による処理から参照可能とする処理を行うことによって、結合処理部 3 の処理に対するインターフェースを提供する（P F インターフェースと称する）。この P F インターフェースは、すでに開発済みの従来の車種のエンジン E C U の要求仕様を満足するように標準化して構成されている。

【0037】

また、結合処理部 3 は、P F インターフェースを介して取得した情報を開発対象のアプリケーションプログラム 4 の要求仕様を満足する情報に変換してアプリケーションプログラム 4 の処理から参照可能とする処理と、アプリケーションプログラム 4 による処理によって生成された要求仕様に従った出力対象の情報を参照して P F インターフェースの要求する情報に変換して P F インターフェースを介して I / O 1 4 へ出力させる処理を C P U 1 1 に行わせるプログラムである。

【0038】

こうした結合処理部 3 を設けることによって、アプリケーションプログラム 4 の開発者は、P F インターフェースの仕様を意識することなく、要求仕様のみを考慮してアプリケーションプログラムを開発することができる。この要求仕様には、入出力するデータのタイミングやデータ型式等の A P インターフェースに関する仕様と、A P インターフェースを介して取得したデータを利用した車両用の制御仕様とが含まれる。制御仕様については、アプリケーションプログラム 2 の開発中であってもたびたび車両メーカー側からの変更の要求があるのに対し、A P インターフェースに関する仕様については、開発の初期段階でほぼフィックスされている。したがって開発の初期段階で A P インターフェースの仕様は決定することができ、また、P F インターフェースの仕様はすでに標準化されているものを流用するため、開発の初期段階から結合処理部 3 の開発を進めることができる。一方、アプリケーションプログラム 2 については、開発の初期段階から要求仕様のみを意識して開発を進めることができ、開発中に制御仕様の変更があったとしても、A P インターフェースは変更する必要がなく、制御仕様に従って制御部分の変更を行うだけで済む。

【0039】

こうした結合処理部 3 を備えたプログラム 1 の詳細な構成の例を図に示して説

明する。

例えば、図3 (a) に示すように、P F インターフェースによってunsigned short型でL S Bの示す値が5 / 6 5 5 3 6 [V] のデータ型式でデータが提供されるよう標準化されている一方、要求仕様のデータ型式がunsigned char型でL S Bの示す値が5 / 2 5 6 [V] の場合について説明する。

【0040】

この場合、図3 (b) に示すように、結合処理部3は、プラットフォームプログラム2による処理から受けとったデータのデータ型及びL S Bの対応する値についての変換処理をCPU11に実行させるように構成する。

すなわち、結合処理部3は、P F インターフェースを介してunsigned short型でL S Bの示す値が5 / 6 5 5 3 6 [V] のデータを取得し、取得したデータをunsigned char型でL S Bの示す値が5 / 2 5 6 [V] のデータに変換（加工）する変換処理を行い、変換したデータをアプリケーションプログラム4による処理から参照可能にする処理をCPU11が実行するための命令コードを備える。例えば結合処理部3による処理では、変換したデータを予め定めたアドレスのRAM13に格納しておき、アプリケーションプログラム4による処理では、このアドレスのRAM13の値を取得するようにして、A P インターフェースを構成し、アプリケーションプログラム4による処理による要求仕様に沿ったデータの参照を行うことができる。

【0041】

従来の車両用制御プログラムの構成では、図4に示すように、プラットフォームプログラム2による処理によって提供されるunsigned short型でL S Bの示す値が5 / 6 5 5 3 6 [V] のデータ型式のデータをアプリケーションプログラムによる処理で取得して、アプリケーションプログラムによる処理で、要求仕様を満たすデータ型式であるunsigned char型のL S Bの示す値が5 / 2 5 6 [V] のデータに変換する変換処理を行う必要があった。したがって、このような変換処理をアプリケーションプログラム内で行う必要があり、アプリケーションプログラムの開発者は、この点を常に意識しながらアプリケーションプログラムを開発する必要があるため、アプリケーションプログラムの開発者の負担が大きかつ

た。しかし、上述した結合処理部 3 を備えることにより、アプリケーションプログラム 4 の開発者は、P F インターフェースのデータ型式を意識することなく、単に要求仕様のみを考慮してアプリケーションプログラム 4 の開発を行うことができる。

【0042】

次に、結合処理部 3 による処理で、データの取得タイミングを調整する例を示す。

図 5 (a) に示すように、プラットフォームプログラム 2 による処理で提供される P F インターフェースは、結合処理部 3 による処理から参照可能なデータを 1 m s 毎に更新する構成で標準化されており、一方、要求仕様としては、8 m s 毎に利用する仕様である。但し、この 8 m s 毎に利用するデータは、16 m s 毎に取り込んだデータとする。

【0043】

この場合、図 5 (b) に示すように、プラットフォームプログラム 2 は、1 m s 毎に水温センサ 23 の電圧値を A/D 変換器 15 を介してプラットフォームプログラム 2 の管理下の R A M 13 に取り込む処理を C P U 11 に実行させるための命令コードを備える。またプラットフォームプログラム 2 は、結合処理部 3 による処理からの要求に応じてプラットフォームプログラム 2 による管理下の R A M 13 に取り込んだ電圧値を結合処理部 3 による処理へ渡す処理を C P U 11 に実行させるための命令コードを備える。

【0044】

そして、結合処理部 3 は、16 m s 毎にプラットフォームプログラム 2 による処理に対して電圧値を要求して、プラットフォームプログラム 2 による処理における管理下のメモリに取り込まれた電圧値を受け取り、結合処理部 3 の処理による管理下のメモリに記憶する処理を C P U 11 に実行させるための命令コードを備える。また、アプリケーションプログラム 4 による処理からの要求に応じて、この結合処理部 3 による処理の管理下のメモリに記憶された電圧値をアプリケーションプログラム 4 による処理へ渡す処理を行うための命令コードを備えることで、A P インターフェースを提供する。

【 0 0 4 5 】

そして、アプリケーションプログラム 4 は、8 m s 毎に結合処理部 3 に対して電圧値を要求し、結合処理部 3 による処理から電圧値を受け取って、利用する処理を行う命令コードを備える。

なお、アプリケーションプログラム 4 による処理からこの結合処理部 3 による処理の管理下のメモリを直接参照して取得する A P インターフェースとしてもよい。

【 0 0 4 6 】

また、プラットフォームプログラム 2 には、結合処理部 3 を 1 6 m s 以内の間隔で実行し、アプリケーション 4 を 8 m s 以内の間隔で実行するための命令コードを備える。

こうした車両用制御プログラム 1 を C P U 1 1 が実行すると、図 5 (b) に示すように、結合処理部 3 による処理で、1 6 m s 毎に電圧値を P F インターフェースを介して取得して記憶しておき、アプリケーションプログラム 4 の処理によって 8 m s 毎に、この記憶している電圧値を利用する。

【 0 0 4 7 】

このように、結合処理部 3 を設けることで、アプリケーションプログラム 4 は、要求仕様を満足する 1 6 m s 毎にサンプリングされた電圧値を、8 m s 毎に利用することができる。すなわち、アプリケーションの開発者は、図 6 に示す従来例のようにアプリケーションプログラムによる処理内でサンプリングタイミング変換しなければならないことを意識する必要はなく、アプリケーションプログラム 4 を開発することができる。

【 0 0 4 8 】

次に、割り込み処理に適用した例を説明する。図 7 に示すように、エンジン回転センサ 2 1 からの信号パルスの立ち下がり、C P U 1 1 に I / O 1 4 から割り込みがかかり、C P U 1 1 はプラットフォームプログラム 2 に備えた割り込み処理ルーチンを実行する。このプラットフォームプログラム 2 の割り込み処理ルーチンは、この割り込み処理ルーチンの前回の実行時からの経過時間をタイマ 1 6 の値を取得して算出し、これに基づいてエンジン回転数を求めて、プラットフ

ォームプログラム 2 の管理下の R A M 1 3 のメモリ領域へ記憶する処理を行うための C P U 1 1 の命令コードを備える。また、プラットフォームプログラム 2 は、結合処理部 3 による処理からの要求に応じてこのプラットフォームプログラム 2 の管理下の R A M 1 3 のメモリ領域へ記憶された回転数を結合処理部 3 による処理に渡す処理を行うための C P U 1 1 の命令コードを備える。そして、プラットフォームプログラム 2 は、結合処理部 3 のタスク（特許請求の範囲における取得プログラムに相当する）を起動する命令コードを備え、この命令コードを C P U 1 1 が実行することで、結合処理部 3 のタスクが起動される。

【0049】

このタスクは、プラットフォームプログラム 2 の処理に回転数を要求するための命令コードと、プラットフォームプログラム 2 による処理から受け取った回転数を結合処理部 3 の管理下の R A M 1 3 のメモリ領域へ記憶する処理を行うための命令コードと、アプリケーションプログラム 4 による処理から回転数の要求があった場合に、この結合処理部 3 の管理下の R A M 1 3 のメモリ領域へ記憶された回転数をアプリケーションプログラム 4 による処理へ渡す処理を行うための命令コードとを備える。

【0050】

そして、エンジン回転数を利用して要求仕様を満たす機能を実現させるためのプログラムであるアプリケーションプログラム 4 の回転数利用処理部は、エンジン回転数を参照する各部で、結合処理部 3 による処理に対して回転数を要求するための命令コードを備える。

【0051】

そして、割り込み処理部の実行レベルは他のプログラム部分よりレベルを高く設定し、結合処理部 3 の実行レベルはアプリケーションプログラム 4 の実行レベルと同一かそれ以下に設定する命令コードをプラットフォームプログラム 2 に備える。そして、C P U 1 1 は、事前にこの実行レベルの設定をするプログラムの命令コードを実行しておく。また、プラットフォームプログラム 2 には、実行レベルに応じて実行すべきタスクを決定する処理を行う命令コードを備える。

【0052】

このような構成によれば、図7に示すように、エンジン回転センサ信号のパルスの立ち下がり、CPU11がプラットフォームプログラム2の割り込み処理ルーチンを実行し、エンジン回転数を求めて、プラットフォームプログラム2の管理下のRAM13のメモリ領域へ記憶する。そして、プラットフォームプログラム2による処理によって、アプリケーションプログラム4が実行中でない場合には、図7に示す割り込み処理P1の後のように、結合処理部3のタスク処理部C1が実行されてプラットフォームプログラムによる処理に対して回転数を要求し、プラットフォームプログラムによる処理では結合処理部3による処理に対して回転数を渡す。結合処理部3のタスク処理部C1の処理により回転数を受け取り、結合処理部3の管理下のRAM13のメモリ領域へ記憶する。そしてプラットフォームプログラム2は、アプリケーションA1を実行する。

【0053】

アプリケーションA1において、回転数を参照する場合には、結合処理部3による処理に回転数を要求する。結合処理部3による処理では、結合処理部3の管理下のRAM13のメモリ領域へ記憶された回転数を渡す。そして、アプリケーションプログラム4の処理では、この回転数を受け取って自動車用の制御処理に利用する。

【0054】

また、図7に示すように、アプリケーションA2の処理中に、エンジン回転センサ信号による割り込みが発生し、プラットフォームプログラム2の割り込み処理P3が実行される場合、割り込み処理P3の完了後、プラットフォームプログラム2によって結合処理部C3が起動されるのであるが、結合処理部C3の実行レベルは、アプリケーションA2の実行レベルと同一かそれ以下に設定されているため、プラットフォームプログラム2による処理では、結合処理部C3の実行を後回しにして、先に、アプリケーションA2の割り込みによる中断時点からの命令を実行する。そのため、アプリケーションA2では、途中で割り込み処理P3を行っているが、回転数を参照する場合には、結合処理部C2の管理下のRAM13のメモリ領域へ記憶された回転数を取得して処理に利用することとなる。

【0055】

このように、アプリケーションプログラム 2 による処理で結合処理部 3 による処理を介して回転数を取得するようにプログラムを構成することで、アプリケーションプログラム 4 の開発者は、プラットフォームプログラム 2 の仕様を意識することなく、単に結合処理部 3 とのやりとりを行う A P インターフェースの仕様を知るだけでアプリケーションプログラム 4 を開発することができる。

【0056】

すなわち、従来は、上述した結合処理部 3 が存在しないため、図 8 (a) のようにアプリケーション A 1 の処理中で、回転数の参照を行うと、プラットフォームの割り込み処理 P 1 でメモリに記憶された回転数がプラットフォームから提供されていた。この場合、アプリケーション A 1 中で複数回のプラットフォームの回転数を参照したとしても、アプリケーション A 1 中では、得られる回転数は全て同じ値となり、問題は生じない。しかし、割り込み処理 P 2 の後にアプリケーション処理 A 2 が実行され、このアプリケーション A 2 中に、エンジン回転センサ信号による割り込みが発生した場合、アプリケーション A 2 の処理は中断され、プラットフォームプログラム 2 による割り込み処理 P 3 が実行される。そして、割り込み処理 P 3 の完了後にアプリケーション処理 A 2 が再開される。この結果、割り込み処理 P 3 中で回転数は新たな値に更新されるので、中断前のアプリケーション処理 A 2 の部分では、プラットフォームから取得した割り込み処理 P 2 による回転数を用いた処理がなされ、割り込み処理 P 3 から復帰した後のアプリケーション処理 A 2 の部分では、プラットフォームから取得した割り込み処理 P 3 による回転数を用いた処理がなされることとなる。このように、1 単位 of アプリケーション処理内で、ハードウェアデバイスからのデータに基づくデータに関して一貫性が保持できない。そのため、従来は、アプリケーションの開発者は、こうしたプラットフォームの制約条件を理解し、例えば図 8 (b) に示すように、割り込みの影響を受けないように、アプリケーション処理の先頭でディスパッチ禁止処理を行い、プラットフォームの A P I を呼び出して回転数を取り込んで、自己の管理下のメモリに記憶する（グローバルコピーすると称する）処理を行い、そして、それ以降のアプリケーション処理では、このグローバルコピーされた回転数を用いて処理を行うようにする必要があった。このように、アプリケ

ーションプログラムの開発者は、プラットフォームの仕様（制約条件）を熟知し、仕様間のギャップを埋めて、アプリケーションプログラムを作成しなければならない。もしも、こうしたプラットフォームの仕様をアプリケーション作成者がよく理解せずに作成すると、例えば図8（a）のように、ときどき動作がおかしくなるといった症状などが生じ、作成したアプリケーションプログラムが、アプリケーション作成者の意図通りに動かない。このような場合、アプリケーションの開発者はプラットフォームの仕様を理解していないため、動作異常の原因を特定するのは困難になってしまう。かといって、複数のアプリケーション（例えばタスク）を動作させる場合、すべてのアプリケーションについてグローバルコピーを行うとメモリ領域の消費や処理時間の消費などオーバーヘッドが大きくなる。そのため、どのアプリケーションが、処理内でのデータの一貫性（同時性）を保てないのかを、プラットフォームの仕様とアプリケーションの仕様とを照らし合わせて判断する必要がある、アプリケーションの開発者の負担が大きく、開発効率を高める上での障害となっていた。

【0057】

しかし、図7に示して説明した本実施例の構成にすれば、図8（b）に示すようなグローバルコピーをアプリケーションプログラム中で行う必要もなくなり、どのアプリケーションプログラムについてグローバルコピーをする必要があるのかなどを検討する必要もなくなる。したがって、アプリケーションの生産性を向上させることができる。

【0058】

以上、アプリケーションプログラムによる処理でデータを入力する場合について説明したが、アプリケーションプログラムによる処理で、データを出力する場合にも、従来のように、直接プラットフォームプログラムの提供するインターフェースにアクセスするのではなく、結合処理部3を介して処理を行うとよい。

【0059】

例えば、図9（a）に示すように、アプリケーションプログラム4は、条件判定処理と、演算処理をCPU11が実行するための命令コードを備える。そして、結合処理部3は、プラットフォームプログラム2による処理によって、例えば

所定周期毎に実行される。

【0060】

そして、結合処理部3には、アプリケーションプログラム2による演算処理によって得られたデータを取得する情報取得処理と、情報取得処理によって取得したデータをプラットフォームプログラム2による負荷駆動処理で用いるデータ型に変換する情報変換処理と、プラットフォームプログラム2による負荷駆動処理を実行させるための負荷駆動要求処理とをCPU11が実行するための命令コードを備える。また、プラットフォームプログラム2は、負荷駆動要求があった場合に、負荷駆動処理を行う。

【0061】

例えば、図9(b)に示すように、エンジンを始動して所定時間経過後、エンジン始動時の水温に応じてO2センサヒータ22の制御を行い、要求仕様として周期256msでDuty制御するものであるとする。

この場合、プラットフォームプログラムでは、図9(b)に示すように、O2センサヒータ22を駆動するプラットフォームのAPIをO2H_DutyOut(周期, Duty)とし、周期のデータ型は、unsigned shortでLSBの対応する値は1msとし、Dutyのデータ型はsigned longでLSBの対応する値は1/65536とする。Dutyのデータ範囲は0~1として、それ以外は動作を保証しない。

【0062】

一方、要求仕様は、アプリケーションプログラム4の演算処理によって出力するDutyがfloat型であり、範囲が0~100%で、システムの要求する周期256msでDuty制御であるとする。

このとき、従来は、アプリケーションプログラムでは、図11及び図12に示すように、アプリケーション内の演算処理で出力されたfloat型のDutyを、プラットフォームが指定する型およびLSBの対応する値に変換処理して、プラットフォームのAPIを呼び出すように構成する必要があった。

【0063】

具体的には、アプリケーションプログラムとプラットフォームプログラムとは

それぞれ図 1'2 に示すように、処理を行っていた。すなわち、アプリケーションプログラムは、図 1 2 に示すように条件判定処理、演算処理、負荷駆動要求処理を行い、プラットフォームは、負荷駆動処理を行う。

【0064】

アプリケーションプログラムでは、S 1 1 0 で、エンジン始動後所定時間経過したか否かを判定し、所定時間経過した場合には（S 1 1 0：成立）、S 1 2 0 へ移行して、所定の処理（始動時水温補間テーブルによる補間）によってD u t y _ A P を求める。一方、エンジン始動後所定時間経過していない場合には（S 1 1 0：不成立）、ヒータ制御は不要であるので、S 1 3 0 へ移行して、D u t y _ A P を 0 % にする。

【0065】

そしてS 1 4 0 では、D u t y _ A P を、プラットフォームのA P I（プラットフォーム命令）の引数として用いられる型とL S Bの示す値に変換してD u t y _ P F として求める。すなわち、変換後のL S Bの示す値が1 / 6 5 5 3 6 となるようにfloat型からsigned long型への変換を行う。続くS 1 5 0 では、D u t y _ P F をプラットフォームの仕様に従って0 ~ 1 でガードする。そして、S 1 6 0 で、周期を2 5 6（ms）、D u t y をD u t y _ P F としてプラットフォーム命令O 2 H _ D u t y を呼び出す。このようにアプリケーションの開発者は、プラットフォームの制約条件を意識して、アプリケーション内に変換処理を設ける必要がある。

【0066】

プラットフォーム側では、この呼び出しに応じて、呼び出し時にセットされた値（引数）に基づき、実際にハードウェアを制御し、ヒータ制御を行う（S 2 0 0）。

このように、プラットフォームプログラムの提供する機能をアプリケーションプログラムから直接利用するため、プラットフォームプログラムの動作上の仕様が、アプリケーションプログラム作成時の制約条件となってしまう。アプリケーションプログラムの開発者は、常にこうした制約条件に注意を払って、入出力のタイミングやデータ形式等に気を配ってこうした制約条件をなくすためのルーチ

ン（例えばCPU11の命令コード）をアプリケーションに組み込む必要があった。

【0067】

しかも、アプリケーションプログラムは、機能の追加、変更等の頻度が高く、こうした制約条件に注意を払いながら、これらの機能の追加や変更を行うのは困難であった。

そこで、本実施例では、図12に示した従来のプログラム構成を図10に示すように変更する。すなわち、上述したように従来は、条件判定処理と、演算処理と、負荷駆動要求処理とをアプリケーションプログラムで行うようにしていたのであるが、本実施例では、図10に示すように結合処理部3に、従来、アプリケーションプログラムによって行っていた図12のS140～160に相当する処理を、図10のS420～440に示すように結合処理部3に設ける。そして結合処理部3には、アプリケーションプログラム4のS310～330によって求められたDutyAPを読み込む処理（S410）を設ける。このようにして図10に示すように、情報加工処理と負荷駆動要求処理を結合処理部3による処理で行う。

【0068】

その結果、アプリケーションプログラム4は、図10のS310～330のように、条件判定処理と演算処理とをCPU11が実行するように構成するだけで済む。

このように、出力処理に関しても、アプリケーションの開発者は、プラットフォームプログラムのインターフェース仕様を気にすることなく、アプリケーションの開発を行うことができる。

【0069】

なお、本実施例では、エンジンECUの例で説明したが、種々の車両用のECUのプログラム開発に適用できる。

【図面の簡単な説明】

【図1】 実施例の車両用制御装置の構成を示すブロック図である。

【図2】 実施例の車両用制御プログラムの構成を示す説明図である。

【図 3】実施例の車両用制御プログラムによるデータ型式の変換処理の例を示す説明図である。

【図 4】従来の車両用制御プログラムによるデータ型式の変換処理の例を示す説明図である。

【図 5】実施例の車両用制御プログラムによるサンプリングタイミングの例を示す説明図である。

【図 6】従来の車両用制御プログラムによるサンプリングタイミングの例を示す説明図である。

【図 7】実施例の車両用制御プログラムによる割り込み処理の実行例を示す説明図である。

【図 8】従来の車両用制御プログラムによる割り込み処理の実行例を示す説明図である。

【図 9】実施例の車両用制御プログラムによるデータの出力処理の例を示す説明図である。

【図 10】実施例の車両用制御プログラムによるデータの出力処理を示すフローチャートである。

【図 11】従来の車両用制御プログラムによるデータの出力処理の例を示す説明図である。

【図 12】従来の車両用制御プログラムによるデータの出力処理を示すフローチャートである。

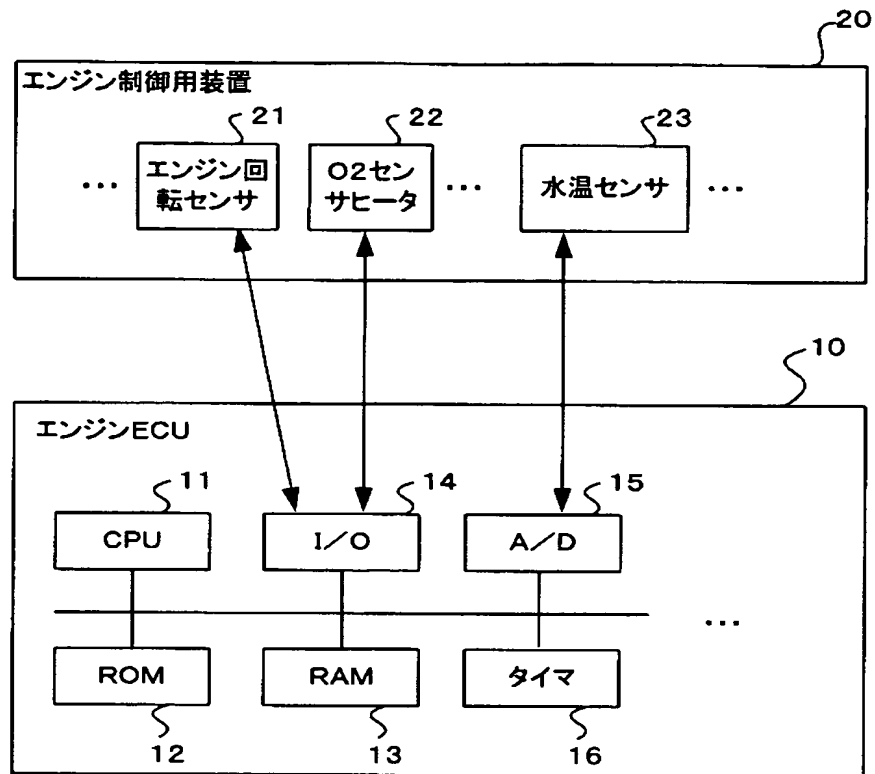
【符号の説明】

- 1…プログラム
- 2…プラットフォーム
- 3…結合処理部
- 4…アプリケーション
- 10…エンジン ECU
- 11…CPU
- 12…ROM
- 13…RAM

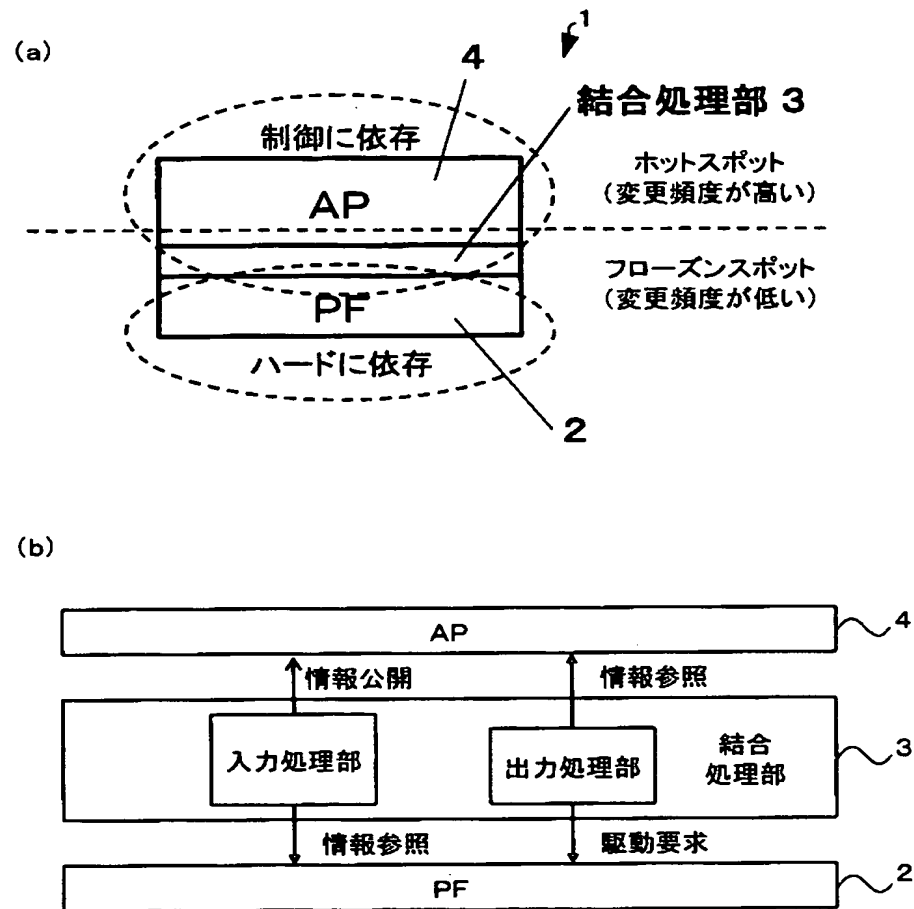
- 1 4 … I / O
- 1 5 … A / D 変換器
- 1 6 … タイマ
- 2 0 … エンジン制御用装置
- 2 1 … エンジン回転センサ
- 2 2 … O₂ センサヒータ
- 2 3 … 水温センサ

【書類名】 図面

【図 1】

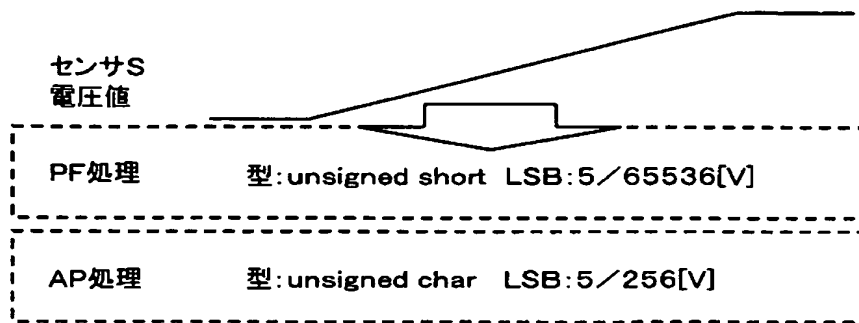


【図 2】

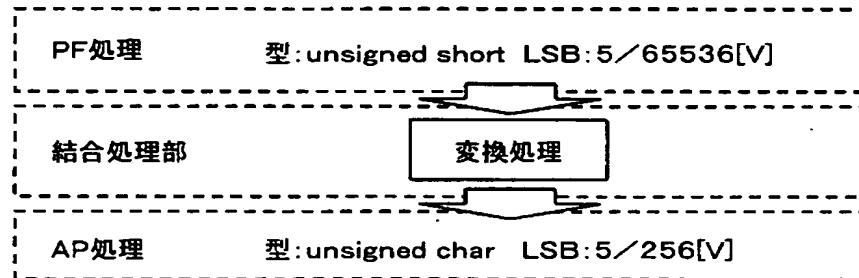


【図 3】

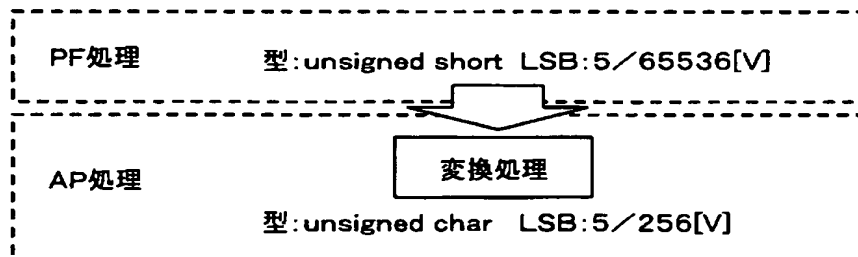
(a)



(b)

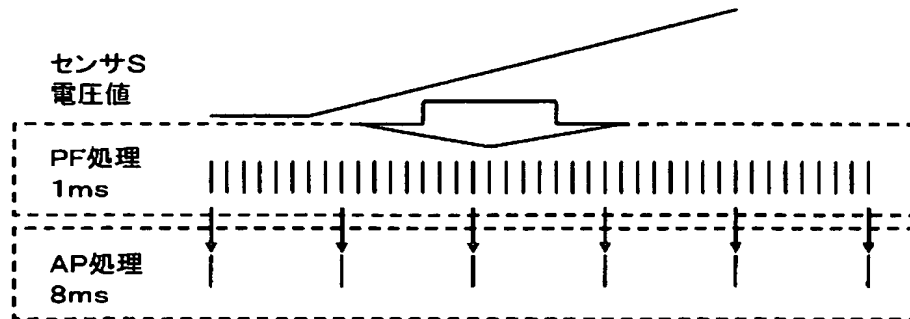


【図 4】

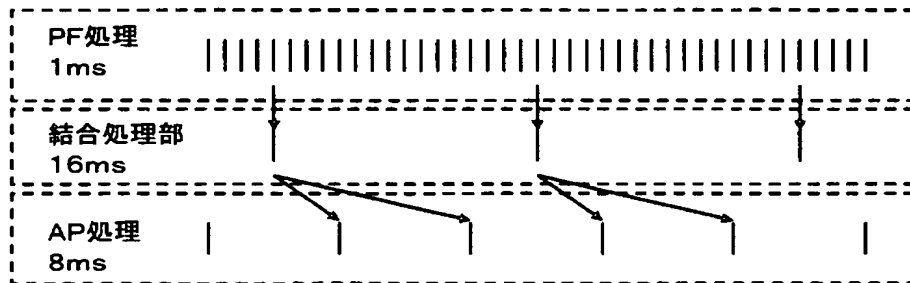


【図 5】

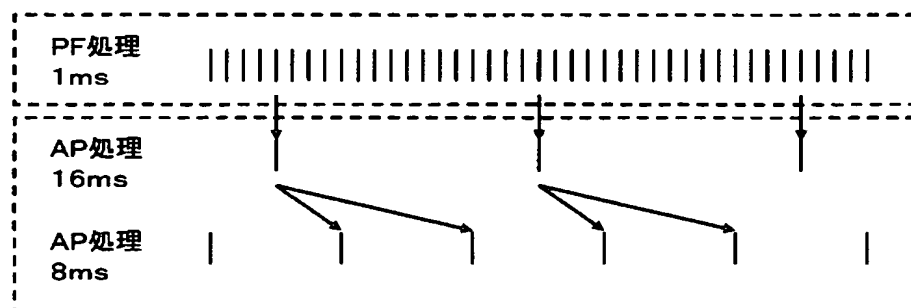
(a)



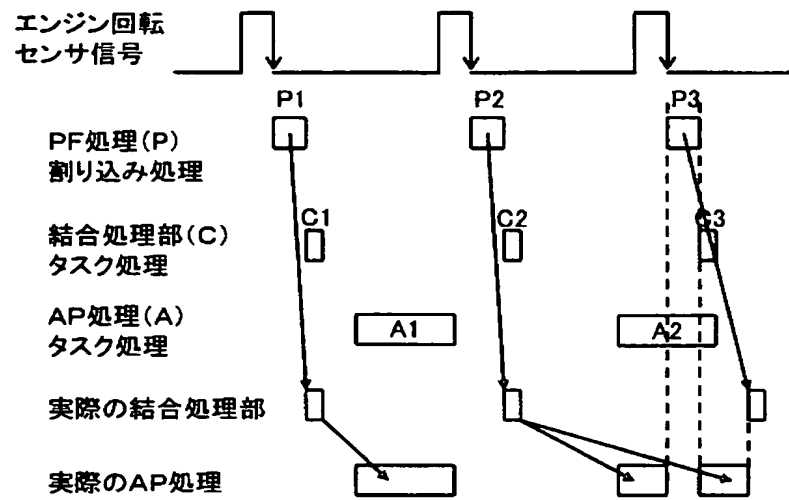
(b)



【図 6】

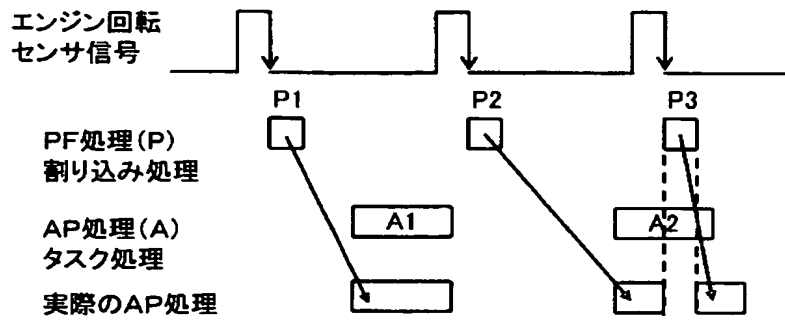


【図 7】

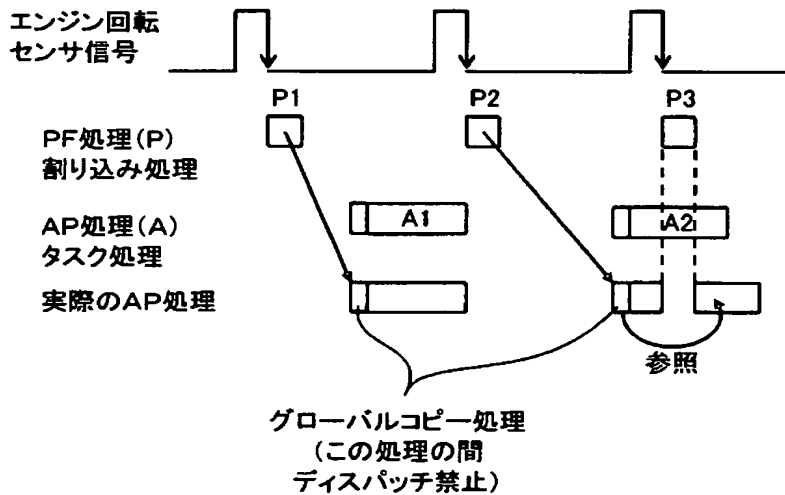


【図 8】

(a)

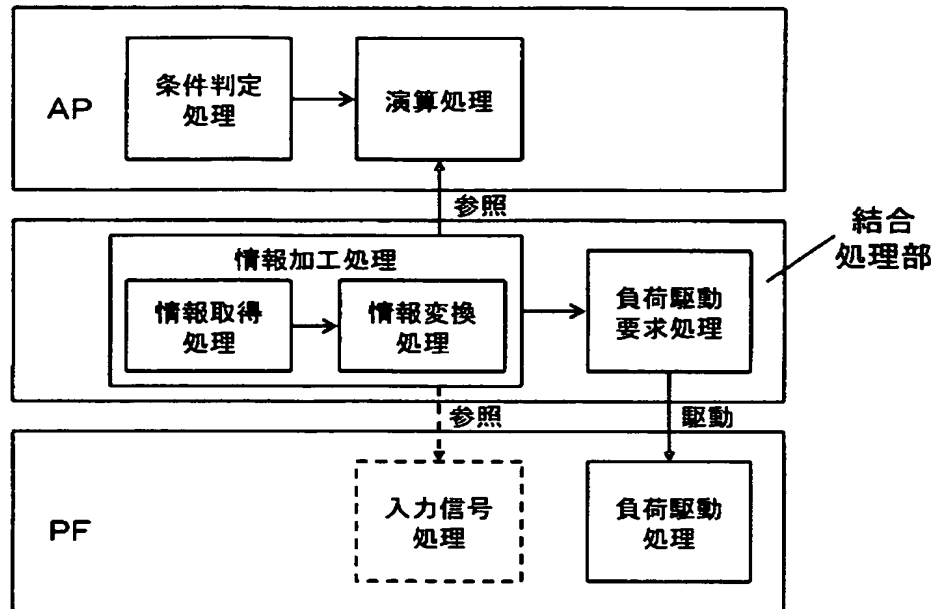


(b)

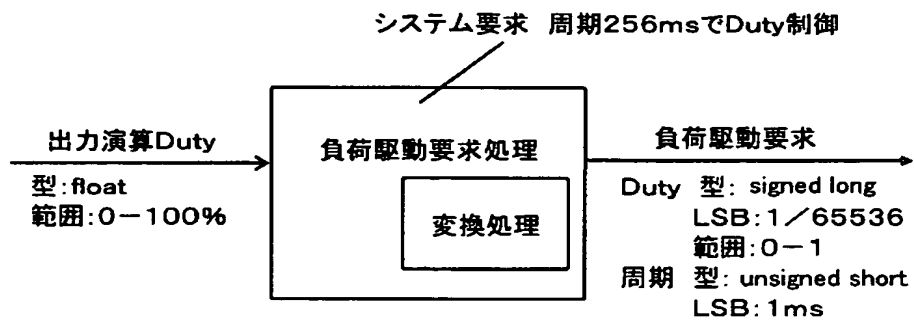


【図9】

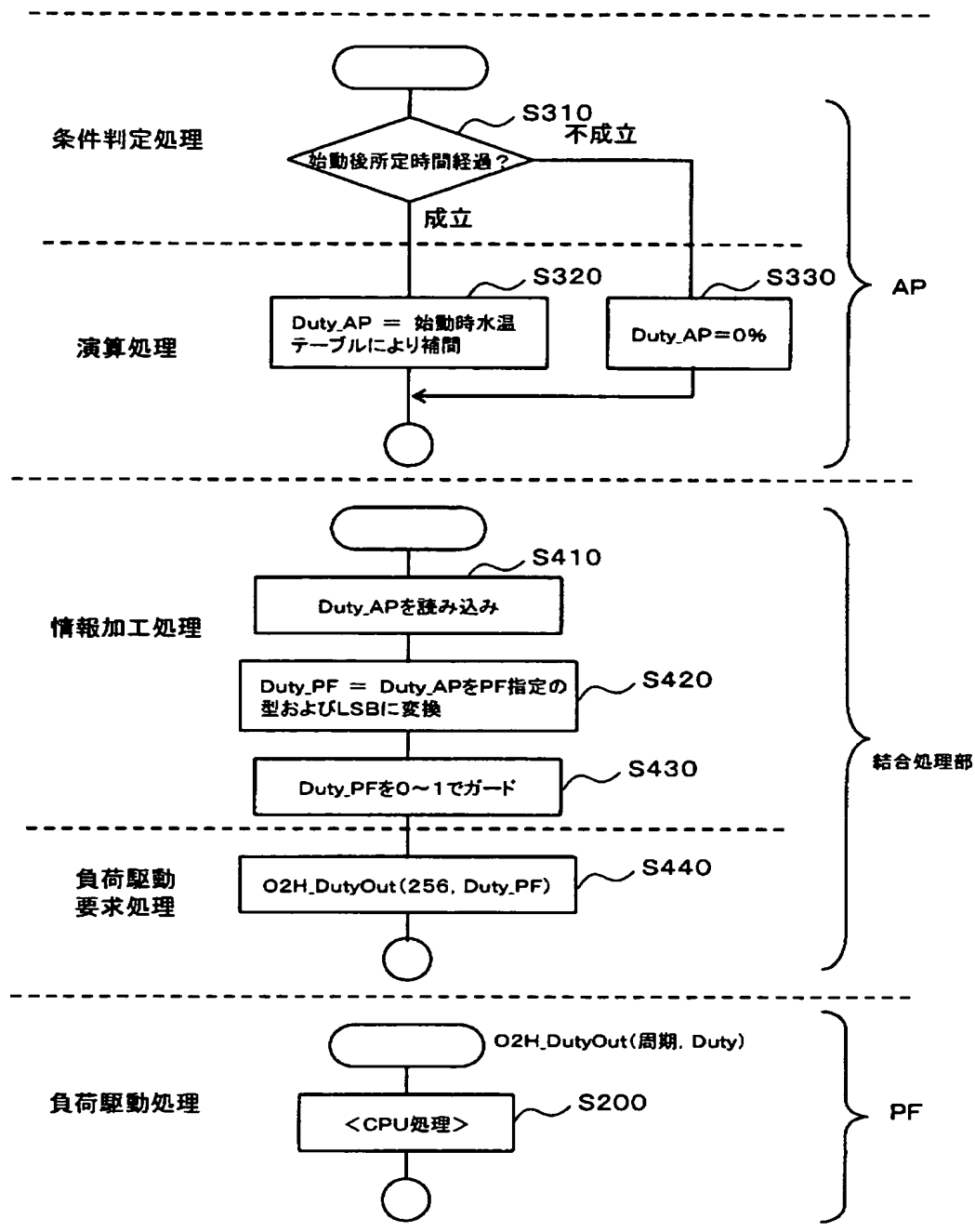
(a)



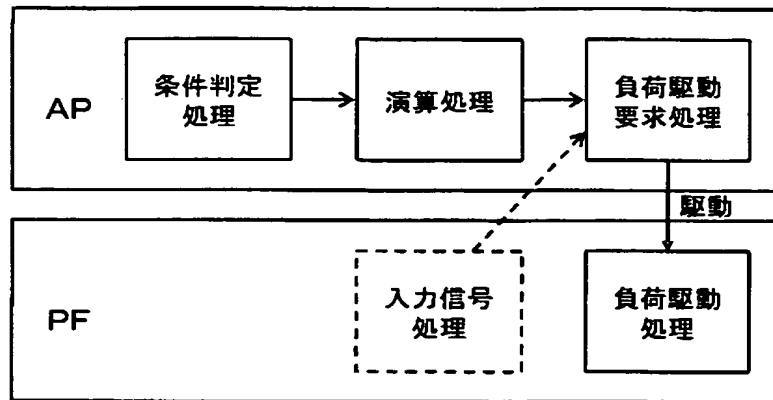
(b)



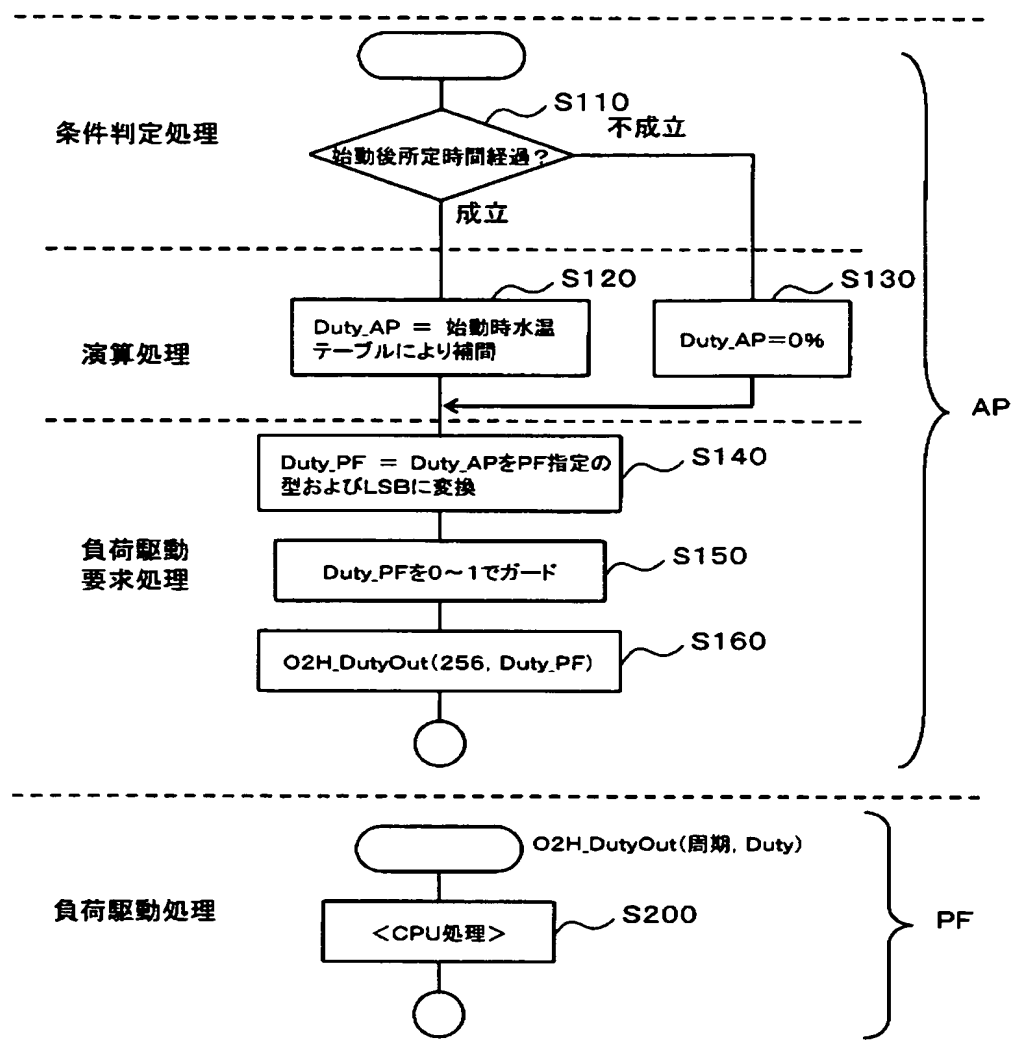
【図10】



【図 11】



【図 12】



【書類名】 要約書

【要約】

【課題】 アプリケーションプログラムの開発者の負担を軽減することのできる車両用制御プログラム等を提供する。

【解決手段】 ハードウェアデバイスからの入力を行い、入力結果に基づくデータを他の処理に渡すための処理をコンピュータに実行させるためのデバイスドライバを含むプログラムであるプラットフォームプログラム 2 と、プラットフォームプログラム 2 による処理によって得られたデータをアプリケーションプログラム 4 による処理に適合するように仲介する処理をコンピュータに実行させるためのプログラムである結合処理部 3 と、プラットフォームプログラム 2 による処理によって得られたデータに基づく処理をコンピュータに実行させるためのプログラムであるアプリケーションプログラム 4 とを備える。

【選択図】 図 2

特願 2 0 0 2 - 3 6 2 4 8 7

出 願 人 履 歴 情 報

識別番号

[0 0 0 0 0 4 2 6 0]

1. 変更年月日

1 9 9 6 年 1 0 月 8 日

[変更理由]

名称変更

住 所

愛知県刈谷市昭和町 1 丁目 1 番地

氏 名

株式会社デンソー